



## **Ice Sheet System model Inverse Methods**

Eric LAROUR<sup>1</sup>, Eric RIGNOT<sup>1,3</sup>, Mathieu MORLIGHEM<sup>1,2</sup>, Hélène SEROUSSI<sup>1,2</sup> Chris BORSTAD<sup>1</sup>, Feras HABBAL<sup>1,3</sup>, Daria HALKIDES<sup>1,4</sup>, Behnaz KHAKBAZ<sup>1</sup>, John SCHIERMEIER<sup>1</sup>, Nicole SCHLEGEL<sup>1</sup>

<sup>1</sup>Jet Propulsion Laboratory - California Institute of Technology

<sup>2</sup>Laboratoire MSSMat, École Centrale Paris, France

<sup>3</sup>University of California, Irvine

<sup>4</sup>Joint Institute for Regional Earth System Science & Engineering, UCLA



Inverse Methods

Larour et al.

Introduction

Model equations

Inverse problem

Algorithm

Synthetic case

Initial state

Inversion

## Outline

### ① Introduction

Model equations

Inverse problem

Algorithm

### ② Synthetic case

Initial state

Inversion

## Inverse Methods

Larour et al.

## Introduction

## Model equations

Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Ice flow model equations

## Field equations

① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu \dot{\boldsymbol{\epsilon}} \quad \mu = \frac{B}{2 \dot{\epsilon}_e^{1-1/n}} \quad (3)$$

## Boundary conditions

Ice/Air interface: Free surface  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_{atm}$   $\mathbf{n} \simeq \mathbf{0}$  on  $\Gamma_s$

Ice/Ocean interface: water pressure  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_w$   $\mathbf{n}$  on  $\Gamma_w$

Ice/Bedrock interface (1): lateral friction  $(\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} = \mathbf{0}$  on  $\Gamma_b$

Ice/Bedrock interface (2): impenetrability  $\mathbf{v} \cdot \mathbf{n} = -\dot{M}_b n_z$  on  $\Gamma_b$

+ Dirichlet condition

## Inverse Methods

Larour et al.

## Introduction

## Model equations

Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Ice flow model equations

## Field equations

① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu \dot{\boldsymbol{\epsilon}} \quad \mu = \frac{B}{2 \dot{\epsilon}_e^{1-1/n}} \quad (3)$$

## Boundary conditions

Ice/Air interface: Free surface  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_{atm}$   $\mathbf{n} \simeq \mathbf{0}$  on  $\Gamma_s$

Ice/Ocean interface: water pressure  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_w$   $\mathbf{n}$  on  $\Gamma_w$

Ice/Bedrock interface (1): lateral friction  $(\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} = \mathbf{0}$  on  $\Gamma_b$

Ice/Bedrock interface (2): impenetrability  $\mathbf{v} \cdot \mathbf{n} = -\dot{M}_b n_z$  on  $\Gamma_b$

+ Dirichlet condition

## Inverse Methods

Larour et al.

## Introduction

## Model equations

Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Ice flow model equations

## Field equations

① Stokes flow:

$$\nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{g} = \mathbf{0} \quad (1)$$

② Incompressibility:

$$\nabla \cdot \mathbf{v} = 0 \quad (2)$$

③ Constitutive law:

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma} + p \mathbf{I} = 2\mu \dot{\boldsymbol{\epsilon}} \quad \mu = \frac{B}{2 \dot{\epsilon}_e^{1-1/n}} \quad (3)$$

## Boundary conditions

Ice/Air interface: Free surface  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_{atm}$   $\mathbf{n} \simeq \mathbf{0}$  on  $\Gamma_s$

Ice/Ocean interface: water pressure  $\boldsymbol{\sigma} \cdot \mathbf{n} = P_w$   $\mathbf{n}$  on  $\Gamma_w$

Ice/Bedrock interface (1): lateral friction  $(\boldsymbol{\sigma} \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{||} = \mathbf{0}$  on  $\Gamma_b$

Ice/Bedrock interface (2): impenetrability  $\mathbf{v} \cdot \mathbf{n} = -\dot{M}_b n_z$  on  $\Gamma_b$

+ Dirichlet condition

## Inverse Methods

Larour et al.

## Introduction

Model equations

## Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknowns
- ex: surface velocities derived from InSAR

## PDE-constrained optimization

Minimize cost function

$$\mathcal{J}(\mathbf{v}, \alpha) = \frac{1}{2} \int_{\Gamma_s} \left( v_x - v_x^{\text{obs}} \right)^2 + \left( v_y - v_y^{\text{obs}} \right)^2 dS + \mathcal{R}(\alpha) \quad (4)$$

Subject to:

$$\begin{aligned}
 \nabla \cdot \mu (\nabla \mathbf{v} + \nabla \mathbf{v}^T) - \nabla p + \rho \mathbf{g} &= \mathbf{0} && \text{in } \Omega \\
 \nabla \cdot \mathbf{v} &= 0 && \text{in } \Omega \\
 \sigma \cdot \mathbf{n} &= \mathbf{f} && \text{on } \Gamma_s \cup \Gamma_w \\
 (\sigma \cdot \mathbf{n} + \alpha^2 v)_{\parallel} &= 0 && \text{on } \Gamma_b \\
 \mathbf{v} \cdot \mathbf{n} &= -\dot{M}_D n_z && \text{on } \Gamma_b
 \end{aligned} \quad (5)$$



## Inverse Methods

Larour et al.

## Introduction

Model equations

## Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Inverse problems

- Basal friction and ice hardness are difficult to measure
- Use extra datasets to infer unknowns
- ex: surface velocities derived from InSAR

## PDE-constrained optimization

Minimize cost function

$$\mathcal{J}(\mathbf{v}, \alpha) = \frac{1}{2} \int_{\Gamma_s} \left( v_x - v_x^{\text{obs}} \right)^2 + \left( v_y - v_y^{\text{obs}} \right)^2 dS + \mathcal{R}(\alpha) \quad (4)$$

Subject to:

$$\begin{aligned} \nabla \cdot \mu (\nabla \mathbf{v} + \nabla \mathbf{v}^T) - \nabla p + \rho \mathbf{g} &= \mathbf{0} && \text{in } \Omega \\ \nabla \cdot \mathbf{v} &= 0 && \text{in } \Omega \\ \sigma \cdot \mathbf{n} &= \mathbf{f} && \text{on } \Gamma_s \cup \Gamma_w \\ (\sigma \cdot \mathbf{n} + \alpha^2 \mathbf{v})_{\parallel} &= \mathbf{0} && \text{on } \Gamma_b \\ \mathbf{v} \cdot \mathbf{n} &= -\dot{M}_b n_z && \text{on } \Gamma_b \end{aligned} \quad (5)$$



## Inverse Methods

Larour et al.

## Introduction

Model equations

Inverse problem

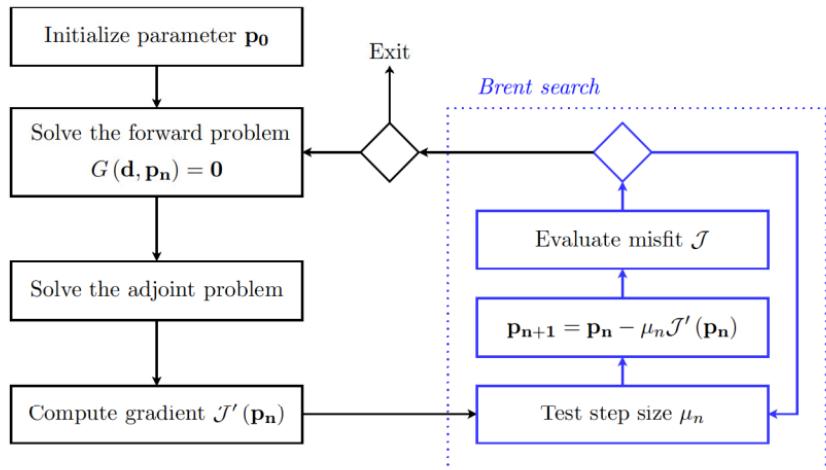
## Algorithm

## Synthetic case

Initial state

Inversion

## Algorithm of resolution



## Inverse Methods

Larour et al.

## Introduction

Model equations

Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

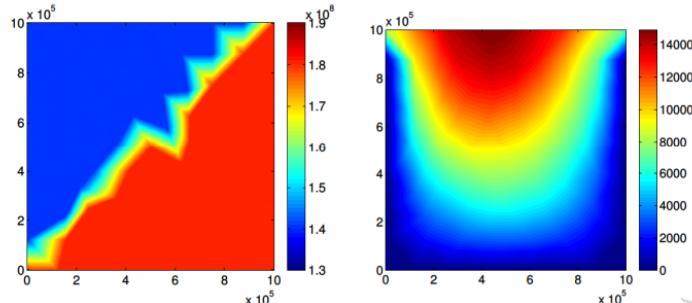
## Synthetic case

```

3      %Generate observation
4      md=model;
5      md=triangle(md,'DomainOutline.exp',100000);
6      md=setmask(md,'all','');
7      md=parameterize(md,'Square.pvt');
8      md=setflowequation(md,'macayeal','all');
9      md.cluster=generic('name',oshostname,'np',2);
10     md=solve(md,DiagnosticSolutionEnum);
```

```

25    disp('      creating flow law paramter');
26    md.materials.rheology_B=1.8*10^8*ones(md.mesh.numberofvertices,1);
27    md.materials.rheology_B(find(md.mesh.x<md.mesh.y))=1.4*10^8;
28    md.materials.rheology_n=3*ones(md.mesh.numberofelements,1);
```



## Inverse Methods

Larour et al.

## Introduction

Model equations

Inverse problem

Algorithm

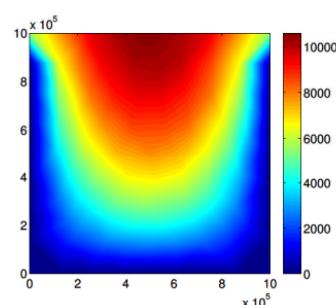
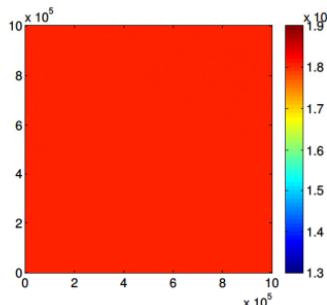
## Synthetic case

Initial state

Inversion

## Start from constant hardness

```
16 %Modify rheology, now constant
17 loadmodel('modell.mat');
18 md.materials.rheology_B(:)=1.8*10^8;
19
20 %results of previous run are taken as observations
21 md.inversion.vx_obs=md.results.DiagnosticSolution.Vx;
22 md.inversion.vy_obs=md.results.DiagnosticSolution.Vy;
23 md.inversion.vel_obs=md.results.DiagnosticSolution.Vel;
24
25 md=solve(md,DiagnosticSolutionEnum);
```



## Inverse Methods

Larour et al.

## Introduction

Model equations

Inverse problem

Algorithm

## Synthetic case

Initial state

Inversion

## Inverse method

```
31 %invert for rheology
32 loadmodel('model2.mat');
33
34 %Set up inversion parameters
35 nsteps=40;
36 md.inversion.iscontrol=1;
37 md.inversion.control_parameters={'MaterialsRheologyBbar'};
38 md.inversion.nsteps=nsteps;
39 md.inversion.cost_functions=101*ones(nsteps,1);
40 md.inversion.cost_functions_coefficients=ones(md.mesh.numberofvertices,1);
41 md.inversion.maxiter_per_step=10*ones(nsteps,1);
42 md.inversion.step_threshold=.8*ones(nsteps,1);
43 md.inversion.gradient_scaling=10^7*ones(nsteps,1);
44 md.inversion.min_parameters=paterson(273)*ones(md.mesh.numberofvertices,1);
45 md.inversion.max_parameters=paterson(200)*ones(md.mesh.numberofvertices,1);
46
47 %Go solve!
48 md=solve(md,DiagnosticSolutionEnum);
```



## Inverse Methods

Larour et al.

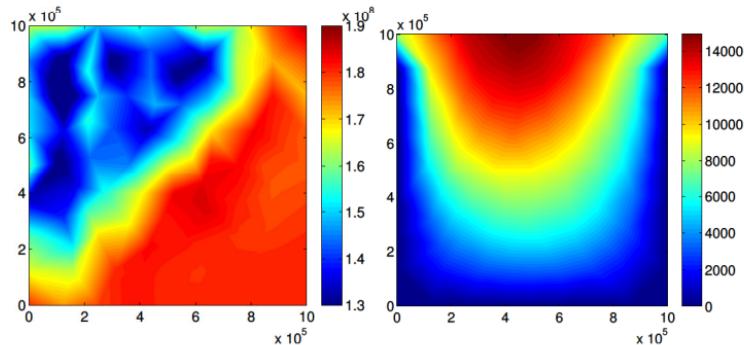
## Introduction

- Model equations
- Inverse problem
- Algorithm

## Synthetic case

- Initial state
- Inversion

## Inverse method



A wide-angle photograph of a desolate, cold landscape, likely an ice field or tundra. In the foreground, the ground is covered in a thick layer of white snow with some dark, irregular patches. Beyond the snow, a range of mountains rises against a clear blue sky. The mountains are heavily covered in snow and ice, with several sharp peaks visible. The lighting suggests a bright, possibly sunny day.

Thanks!